UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/693,589	10/24/2003	Jeffrey P. Snover	MS1-1742US	1106

22801 7590 02/02/2009

LEE & HAYES, PLLC
601 W. RIVERSIDE AVENUE
SUITE 1400
SPOKANE, WA 99201

EXAMINER	
DESAI, RACHNA SINGH	
ART UNIT	PAPER NUMBER
2176	
MAIL DATE	DELIVERY MODE
02/02/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/693,589	SNOVER ET AL.	
	Examiner	Art Unit	
	RACHNA S. DESAI	2176	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *17 November 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-37* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-12 and 20-37* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *12/11/08*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to communications: A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 11/17/08 has been entered.

2. Claims 1-37 are pending. Claims 1, 12, 20, and 26 are independent claims. Claims 1, 12, 20, and 26 have been amended.

Claim Rejections - 35 USC § 112

3. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

4. Claims 1-12 and 20-37 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

Applicant has amended claims to recite, ***executing a pipeline of commands, wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular command separated from each other particular command by a delimiter and executed serially.*** There does not appear to be adequate support for this limitation in the current specification. Correction and/or clarification is requested.

Allowable Subject Matter

5. Claim 12 would be allowable if rewritten or amended to overcome the rejection(s) under 35 U.S.C. 112, 1st paragraph, set forth in this Office action.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Applicant has provided evidence in this file showing that the invention was owned by, or subject to an obligation of assignment to, the same entity as Muhlestein et al. at the time this invention was made, or was subject to a joint research agreement at the

time this invention was made. However, reference Muhlestein additionally qualifies as prior art under another subsection of 35 U.S.C. 102, and therefore, is not disqualified as prior art under 35 U.S.C. 103(c).

Applicant may overcome the applied art either by a showing under 37 CFR 1.132 that the invention disclosed therein was derived from the invention of this application, and is therefore, not the invention "by another," or by antedating the applied art under 37 CFR 1.131.

7. Claims 1-11 and 20-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Muhlestein et al., US 2003/0018765 A1 (field 01/23/03) in view of Varian, Meilinda, "Plunging into Pipes", SHARE Europe AM91, pages 1087-1110, October 1991 (as supplied by the IDS).

Regarding claim 1, Muhlestein dislcoses a system and method for accessing management functionality through a command line utility. Muhlestein discloses a set of commands for the WMI command utility configured by an underlying object model command schema. The object-oriented command schema defines the command line utility comprising a plurality of commands which meets the limitation ***a pipeline comprising a plurality of object-based commands***. See pages 1 and 2, paragraphs [0011]-[0013].

The command line interface permits the entry of command and control functions that are based on and operate against a target WMI schema exposed through the WMI

infrastructure which represents the systems, applications, networks, and other managed components of a target system using an alias object. The command line utility executes an alias object which is a command in order to facilitate a specific administrative task (i.e. method or process). The command schema drives the WMI command line utility and defines the commands used in the utility. An example method for implementing the WMI command utility begins when a command is entered into a command line and actually received by an executable file within the WMI command line utility. The utility, through the executable file, performs a series of operations on the command. The utility interprets the command based on the definition and executes the command as a series of WMI API calls where the WMI data (parseable object) retrieved through API calls is transformed into XML information that is readable by the WMI command line utility which meets the limitations, ***receiving a parseable object emitted from a prior object-based command within the pipeline comprising a plurality of object-based commands, the prior object-based command being one of the plurality of object-based commands*** and ***wherein the parseable object includes at least one method***. The WMI data is returned in XML to the command line utility. See pages 5 and 9.

Muhlestein system *supports* a pipeline of multiple commands. The command line utility executes an alias object which is a command in order to facilitate a specific administrative task (i.e. method or process). The WMI input is provided to a command that outputs the results which meets the limitation, ***operating environment that supports the pipeline of a plurality of object-based commands is configured to support execution of object based commands within the same process***.

The command line utility allows a user to tailor commands. Properties of a command can be arranged in named formats that include property values that can be formatted for display according to a specific presentation strategy which meets the limitation, ***obtaining a data type for the parseable object***. See pages 5 and 9-10. The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias which meets the limitation, ***obtaining format information describing a format for the data type***. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user which meets the limitations, ***such that a subsequent object-based command within the pipeline which receives the parseable object is configured to communicate with the prior object-based command within the pipeline through the parseable object emitted from the prior object-based command*** and ***emitting a format object for access by another subsequent object-based command, the format object being based on the format information***. See pages 5 and 9-10.

Examiner note: The subsequent object-based command can be an output command configured to render the results of the pipeline based on the received parseable object and format object. In Muhlestein, the WMI data (parseable object) and format information is used to present the WMI information.

Muhlestein does not teach ***executing a pipeline of commands, wherein a pipeline is a plurality of commands entered as a single command string on a***

command line, each particular command separated from each other particular command by a delimiter and executed serially.

However, Varian discloses a CMS pipelines for processing data. Varian discloses a pipeline concept in which a plurality of commands are hooked together to form a pipeline and are separated in a line with stage separator characters which is usually a vertical bar which meets the limitation, ***executing a pipeline of commands, wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular command separated from each other particular command by a delimiter and executed serially***. See pages 1087-1088, "The Pipeline Concept"—"A CMS Pipelines Primer".

It would have been obvious to a person of ordinary skill in the art at the time of the invention to have incorporated Varian's single command string on a command line within the command line system of Muhlestein because using a single command string to break an existing program into a number of small stages that can be called using a single command string was known in the art. See pages 1087-1088 of Varian which discusses the "Pipeline Concept". Thus, using the known technique for breaking down a program into smaller steps in a pipeline would have been obvious to a person of ordinary skill in the art at the time of the invention.

In reference to claim 2, Muhlestein teaches the utility interprets the command based on the definition and executes the command as a series of WMI API calls where the WMI data (parseable object) retrieved through API calls is transformed into XML

information that is readable by the WMI command line utility. See page 10, paragraph [0095].

In reference to claim 3, Muhlestein teaches the WMI data (parseable object) and format information is used to present the WMI information. See pages 5 and 9-10. The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

In reference to claim 4, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string which meets the limitation, ***the rendering of results comprises displaying on a console.*** See page 6, paragraph [0063]. Muhlestein further discloses displaying the formatted data through a graphical user interface. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claim 5, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string. Any display strategies to be used in displaying properties will be defined by the XSL style sheet in formatting displays. See page 6, paragraph [0063]. Muhlestein further discloses displaying the formatted data through a graphical user interface. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claim 6, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string. Any display strategies to be used in displaying properties will be defined by the XSL style sheet in formatting displays. See page 6, paragraph [0063]. Muhlestein further discloses displaying the formatted data through a graphical user interface which meets

the limitation, ***rendering of the results comprises displaying in a graphical user interface***. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claim 7, Muhlestein teaches the command schema comprises an alias class defining a command template and a format class as a subclass to the alias class, each instance of the format class representing a list of properties that will be returned through processing an alias. The alias object is a command that is executed in order to capture the features of the target class and to facilitate a task. Alias objects are instances of command-related classes organized into a command schema. The command schema defines the commands and the command line utility uses the aliases to interpret command information and apply the interpretation against the target schema. Properties of the alias object can be arranged in named formats that include property values that can be formatted for display. See page 5. The formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches permitting the generation of additional commands to be added to one or more commands. The user can parameterize commands by specifying locations of command definitions and target system objects. A connection class, a subclass to the alias class, defines the connection instances, each connection instance representing connection parameters used by an alias to establish a connection to target namespace within the target schema. The format class represents a list of properties that will be returned through

processing an alias. The command schema is extensible to permit the generation of additional commands to be added to the set of commands. See page 8 and page 34.

Regarding claim 8, The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

Regarding claim 9, Muhlestein teaches the WMI data is transformed into XML information. The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

Regarding claim 10, Muhlestein teaches the WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

In reference to claim 11, Muhlestein teaches properties of the alias object can be arranged in named formats that include property values that can be formatted for display. See page 5. The formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059].

Regarding **claims 20-25**, claims 20-25 are drawn to a system comprising the hardware to carrying out the steps of claim 1. Thus claims 20-25 are rejected under the same rationale used in claims 1-2, 11, 7-8, and 10 respectively above.

In reference to claims 26, Muhlestein teaches a command line utility comprising an object model command schema to define a mapping between one or more commands an object model target schema, the one or more commands generated by the command schema and configured to operate against the target schema through the command line. Compare to ***"a method for providing a data driven command line output"***. The command line utility comprises an alias class defining a command template, each alias of the alias class representing a single command and a format class as a subclass to the alias class, each instance of the format class representing a list of properties that will be returned through processing an alias. The command schema defines the commands and the command line utility uses the aliases to interpret command information and apply the interpretation against the target schema. See page 2, page 6, paragraphs [0054]-[0059], and page 34. For example, a command is received through an interface by the WMI command line utility. The utility interprets

the command based on its definition. The command is executed as a series of API calls against a target namespace. The WMI infrastructure at the target station then performs the WMI operations against the target object. The WMI data retrieved is transformed at operation into XML information that is readable by the command line utility. See page 9, paragraph [0091]-page 10, paragraph [0095]. Compare to ***"receiving command-line instruction containing an output command configured to receive at least one object, the object having at least one method; and executing the output command to manipulate at least one object"***.

The command line interface permits the entry of command and control functions that are based on and operate against a target WMI schema exposed through the WMI infrastructure which represents the systems, applications, networks, and other managed components of a target system using an alias object. The command line utility executes an alias object which is a command in order to facilitate a specific administrative task (i.e. method or process). The command schema drives the WMI command line utility and defines the commands used in the utility. An example method for implementing the WMI command utility begins when a command is entered into a command line and actually received by an executable file within the WMI command line utility. The utility, through the executable file, performs a series of operations on the command. The utility interprets the command based on the definition and executes the command as a series of WMI API calls where the WMI data (parseable object) retrieved through API calls is transformed into XML information that is readable by the WMI command line utility which meets the limitations, ***wherein the receiving occurs as part of a pipeline of a***

plurality of object-based commands and ***wherein the parseable object includes at least one method***. The WMI data is returned in XML to the command line utility. See pages 5 and 9. The command line utility allows a user to tailor commands. Properties of a command can be arranged in named formats that include property values that can be formatted for display according to a specific presentation strategy. See pages 5 and 9-10. The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user which meets the limitation, ***such that a subsequent object-based command within the pipeline which receives the parseable object is configured to communicate with the prior object-based command within the pipeline through the parseable object emitted from the prior object-based command***. See pages 5 and 9-10.

Properties of the alias object can be arranged in named formats that include property values that can be formatted for display. See page 5. The formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Compare to ***"output a result to an output destination"***.

Muhlestein does not teach ***wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular***

command separated from each other particular command by a delimiter and executed serially.

However, Varian discloses a CMS pipelines for processing data. Varian discloses a pipeline concept in which a plurality of commands are hooked together to form a pipeline and are separated in a line with stage separator characters which is usually a vertical bar which meets the limitation, ***wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular command separated from each other particular command by a delimiter and executed serially***. See pages 1087-1088, "The Pipeline Concept"—"A CMS Pipelines Primer".

It would have been obvious to a person of ordinary skill in the art at the time of the invention to have incorporated Varian's single command string on a command line within the command line system of Muhlestein because using a single command string to break an existing program into a number of small stages that can be called using a single command string was known in the art. See pages 1087-1088 of Varian which discusses the "Pipeline Concept". Thus, using the known technique for breaking down a program into smaller steps in a pipeline would have been obvious to a person of ordinary skill in the art at the time of the invention.

In reference to claims 27-28, Muhlestein teaches an object-based command-line environment. See abstract and pages 1-2.

In reference to claim 29, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string which meets the limitation, ***the rendering of results comprises displaying on a console.*** See page 6, paragraph [0063]. Muhlestein further discloses displaying the formatted data through a graphical user interface. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claim 30, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string. Any display strategies to be used in displaying properties will be defined by the XSL style sheet in formatting displays. See page 6, paragraph [0063]. Muhlestein further

discloses displaying the formatted data through a graphical user interface. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claim 31, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the user in multiple formats as determined by the XSL file located through the format string. Any display strategies to be used in displaying properties will be defined by the XSL style sheet in formatting displays. See page 6, paragraph [0063]. Muhlestein further discloses displaying the formatted data through a graphical user interface which meets the limitation, ***rendering of the results comprises displaying in a graphical user interface***. See page 36, claims 34-36 and figures 2 (illustrates a console) and figure 3.

In reference to claims 32, Muhlestein teaches the formats attribute includes a list of properties that are to be displayed using the given format. See page 34 and page 6, paragraphs [0054]-[0059]. Muhlestein further teaches the Format string provides a location of an XSL file that can be used to format the display for the list of properties being returned to the user through the alias. The WMI command line utility returns information as XML documents which are processed using XSL to render reports for the

user in multiple formats as determined by the XSL file located through the format string. Any display strategies to be used in displaying properties will be defined by the XSL style sheet in formatting displays. See page 6, paragraph [0063].

Regarding claim 36, the WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

Regarding claim 37, Muhlestein teaches the WMI data is transformed into XML information. The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

Regarding claim 33, Muhlestein teaches the WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

Regarding claim 34, The WMI command line receives the WMI XML data and applies an XSL style sheet to format the text for display to the user. The XSL style sheet used is based on the XSL file designated through format specifications within the alias. See pages 5 and 9-10. The XSL style sheet is applied to format the display and present the WMI information for display to a user.

In reference to claim 35, Muhlestein further teaches permitting the generation of additional commands to be added to one or more commands. The user can parameterize commands by specifying locations of command definitions and target system objects. A connection class, a subclass to the alias class, defines the connection instances, each connection instance representing connection parameters used by an alias to establish a connection to target namespace within the target schema. The format class represents a list of properties that will be returned through processing an alias. The command schema is extensible to permit the generation of additional commands to be added to the set of commands. See page 8 and page 34

Response to Arguments

8. Applicant's arguments and amendments filed 11/17/08 have been fully considered.

Applicant's amendments have introduced a new rejection under 35 USC 112, first paragraph as there does not appear to be support in the Specification for the newly

added limitation. Correction is required. It is noted claim 12 which was previously allowed, needs to be rewritten or amended to overcome the rejections under 35 USC 112, first paragraph.

Applicant's arguments with respect to claims 1-11 and 20-37 have been considered but are moot in view of the new ground(s) of rejection under 35 USC 103 over Muhlestein in view of Varian. Specifically, the amended portion reciting, ***executing a pipeline of commands, wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular command separated from each other particular command by a delimiter and executed serially*** was not taught by Muhlestein. However, Varian discloses a CMS pipelines for processing data. Varian discloses a pipeline concept in which a plurality of commands are hooked together to form a pipeline and are separated in a line with stage separator characters which is usually a vertical bar which meets the limitation, ***executing a pipeline of commands, wherein a pipeline is a plurality of commands entered as a single command string on a command line, each particular command separated from each other particular command by a delimiter and executed serially***. See pages 1087-1088, "The Pipeline Concept"—"A CMS Pipelines Primer". It would have been obvious to a person of ordinary skill in the art at the time of the invention to have incorporated Varian's single command string on a command line within the command line system of Muhlestein because using a single command string to break an existing program into a number of small stages that can be called using a single command string was known in the art. See pages 1087-1088 of Varian which discusses the "Pipeline

Concept". Thus, using the known technique for breaking down a program into smaller steps in a pipeline would have been obvious to a person of ordinary skill in the art at the time of the invention.

In view of the comments above, the rejections are maintained.

Conclusion

9. Any inquiry concerning this communication or earlier communications from the examiner should be directed to RACHNA S. DESAI whose telephone number is (571)272-4099. The examiner can normally be reached on M-F (8:30AM-6:00PM).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Doug Hutton can be reached on 571-272-4137. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Rachna S Desai/
Primary Examiner, Art Unit 2176
01/25/09